MOTOROLA
intelligence everywhere™

digital dna™

3-Phase AC Induction Motor Drive with Tachogenerator Using MC68HC908MR32

Designer Reference Manual

M68HC08 Microcontrollers

DRM020/D
Rev. 0, 03/2003

MOTOROLA.COM/SEMICONDUCTORS

3-Phase AC Induction Motor Drive with Tachogenerator Using MC68HC908MR32

Designer Reference Manual — Rev 0

by:

Radim Visinka, Ph.D.
Motorola Czech Systems Laboratories
Roznov pod Radhostem, Czech Republic

Revision history

To provide the most up-to-date information, the revision of our documents on the World Wide Web will be the most current. Your printed copy may be an earlier revision. To verify you have the latest information available, refer to:

http://www.motorola.com/semiconductors

The following revision history table summarizes changes contained in this document. For your convenience, the page number designators have been linked to the appropriate location.

Revision history

Date	Revision Level	Description	Page Number(s)
Feb. 2003	1	Initial version	N/A

List of Sections

List of Sections

Table of Contents

Section 1. Introduction

Section 2. System Description

Section 3. Hardware Design

Section 4. Software Design

Section 5. System Setup

Appendix A. References

Appendix B. Glossary

List of Figures

List of Tables

List of Tables

Section 1. Introduction

1.1 Contents

1.2 Application Functionality

This Reference Design describes the design of a 3-phase AC induction motor drive with tachogenerator speed sensor. It is based on Motorola's MC68HC908MR32 microcontroller which is dedicated for motor control applications. The system is designed as a low cost motor drive system for medium power three phase AC induction motors and is targeted for applications in both industrial and appliance fields (e.g. washing machines, compressors, air conditioning units, pumps or simple industrial drives).

The drive runs in a speed closed loop using a speed sensor. The code can easily be modified to run the drive in open loop if it is required by the application.

The drive to be introduced is intended as a reference platform for a 3-phase AC induction motor drive. It can be used as a good starting point for user's own design of his application according to his special requirements. It can save a lot of development engineering time and speed up the time to market.

The Reference Design incorporates both hardware and software parts of the system including hardware schematics and layout with a bill of material, and a software listing.

1.3 Benefits of the Solution

The design of very low cost variable speed 3-phase motor control drives has become a prime focus point for the appliance designers and semiconductor suppliers. Replacing variable speed universal motors by maintenance-free, low noise asynchronous (induction) motors is a trend that supposes total system costs being equivalent. The big push in this direction is given by several factors:

- new regulations dealing with electrical noise in power distribution lines and low power consumption
- the flexibility that can be achieved in using asynchronous motors with variable frequency
- the maturity level and affordable price trend of power devices
- the system efficiency optimization that microprocessor controlled drives can provide
- the size, weight and dissipated power reduction of the motors for a given mechanical power

Section 2. System Description

2.1 Contents

2.2 System Concept

The system is designed to drive a 3-phase AC induction motor. The microcontroller runs the main control algorithm. According to the user interface input and feedback signals, it generates 3-phase PWM output signals for the motor inverter.

For the drive, a standard system concept is chosen (see **Figure 2-1**). The system incorporates the following hardware parts:

- power supply rectifier,
- three-phase inverter,
- optoisolation
- feedback sensors: DC-Bus voltage, DC-Bus current, tachogenerator for motor speed measurement,
- microcontroller MC68HC908MR32.

The drive can basically be controlled in two different ways (or operating modes) that can be set by an on-board jumper.

- In the Manual Operating Mode, the required speed is set by Start/Stop switch, Forward/Reverse switch and speed potentiometer.

- In the Demo Operating Mode, the required speed profile is pre-programmed and the only control input is the “Start” switch.

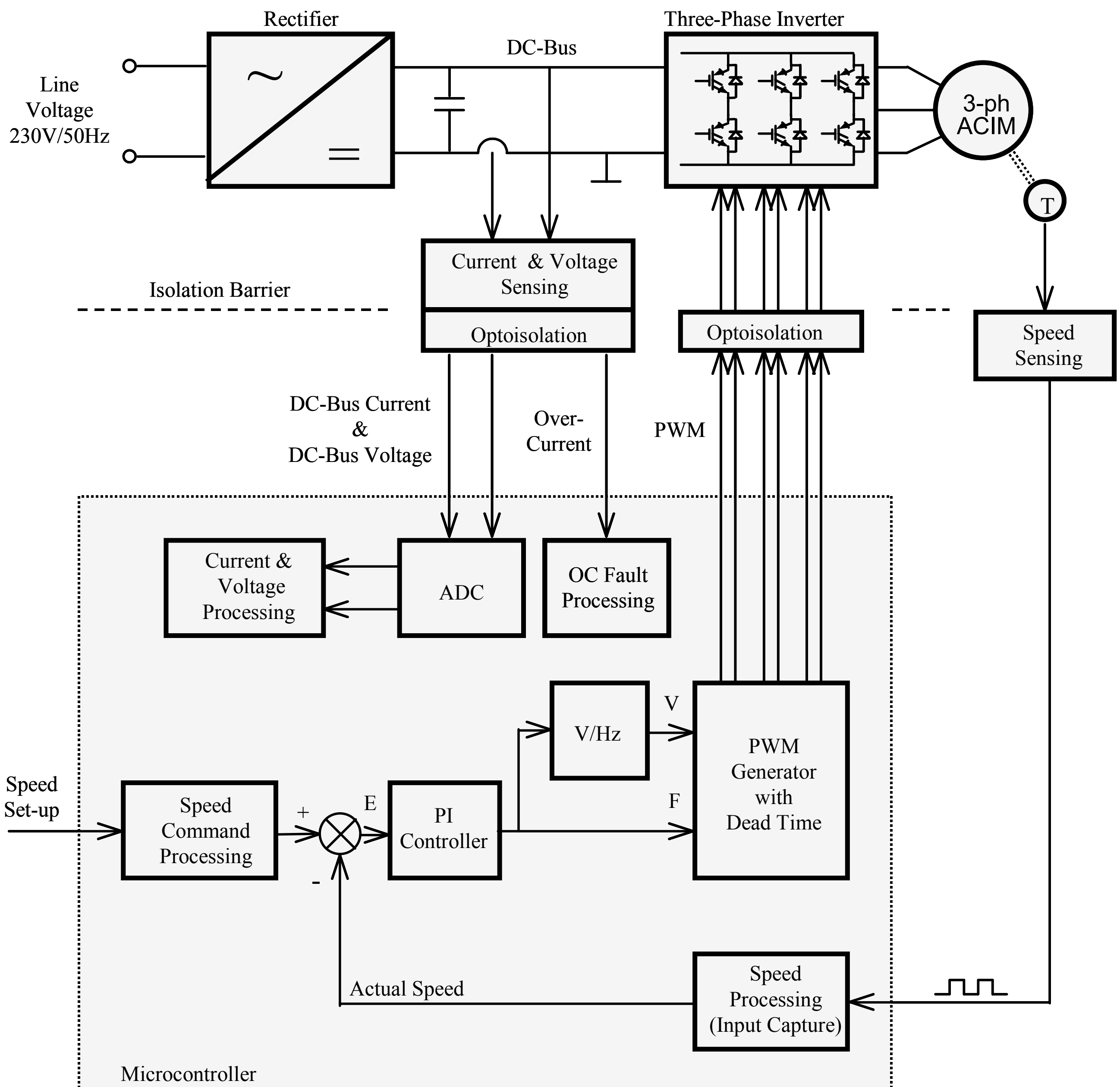

Figure 2-1. System Concept

The **control process** is following:

The state of the sensors is periodically scanned in the software timer loop, while the speed of the motor is calculated utilizing the Input Capture interrupt. According to the operating mode setup and state of the control signals (Start/Stop switch, Forward/Reverse switch, speed potentiometer), the speed command is calculated using an acceleration/deceleration ramp. The comparison between the actual speed command and the tacho speed generates a speed error E. The speed error is brought to the speed PI controller that generates a new corrected motor frequency. Using a V/Hz ramp, the corresponding voltage is calculated. The PWM generation process calculates a system of three-phase voltages with required amplitude and frequency, includes dead time and finally, the 3-phase PWM motor control signals are generated.

The DC-Bus voltage and DC-Bus current are measured during the control process. They are used for over-voltage and over-current protection of the drive. The over-voltage protection is performed by software while the over-current fault signal utilizes a fault input of the microcontroller.

If any of the above mentioned faults occurs, the motor control PWM outputs are disabled in order to protect the drive and fault state of the system is displayed.

WARNING: ***It is strongly recommended to use an opto-isolation (optocouplers and optoisolation amplifiers) during the development time to avoid any damage to the development system.***

2.3 Volt-per-Hertz Control Technique

The drive is designed as a “Volt-per-Hertz“ drive. It means that the control algorithm keeps constant magnetizing current (flux) of the motor by varying the stator voltage with frequency. The commonly used Volt-per-Hertz ramp of a 3-phase AC induction motor illustrates **Figure 2-2**

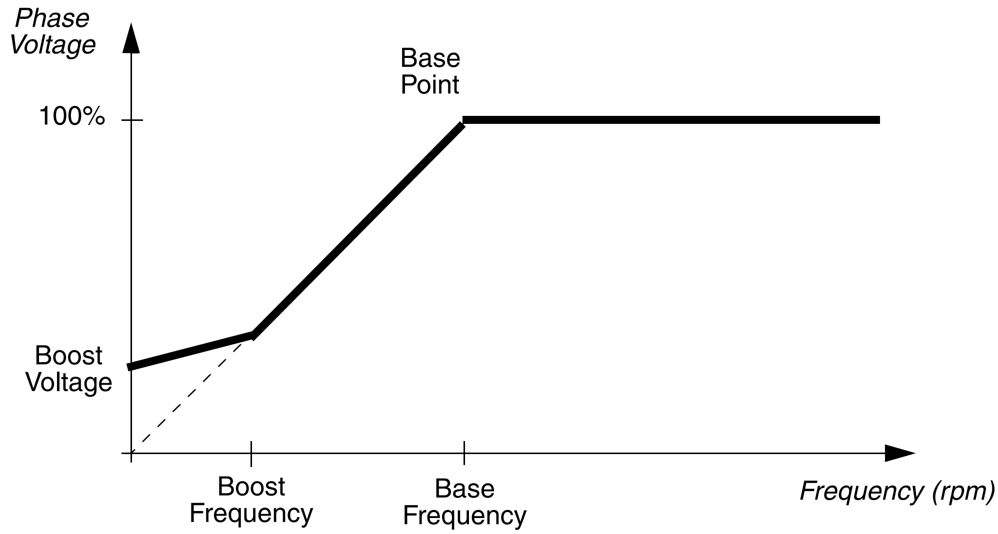

Figure 2-2. Volt-per-Hertz Ramp

The Volt per Hertz ramp is defined by the following parameters:

- Base Point - defined by Base Frequency (usually 50Hz or 60Hz)
- Boost - Defined by Boost Voltage and Boost Frequency

The ramp profile is set to the specific motor and can be easily changed to accommodate different ones.

Section 3. Hardware Design

3.1 Contents

3.2 System Configuration

The application is designed to drive the 3-phase AC motor. It consists of the following modules (see **Figure 3-1**):

- MC68HC908MR32 control board
- 3-phase AC/BLDC high voltage power stage
- Optoisolation board
- 3-phase AC induction motor with speed tachogenerator

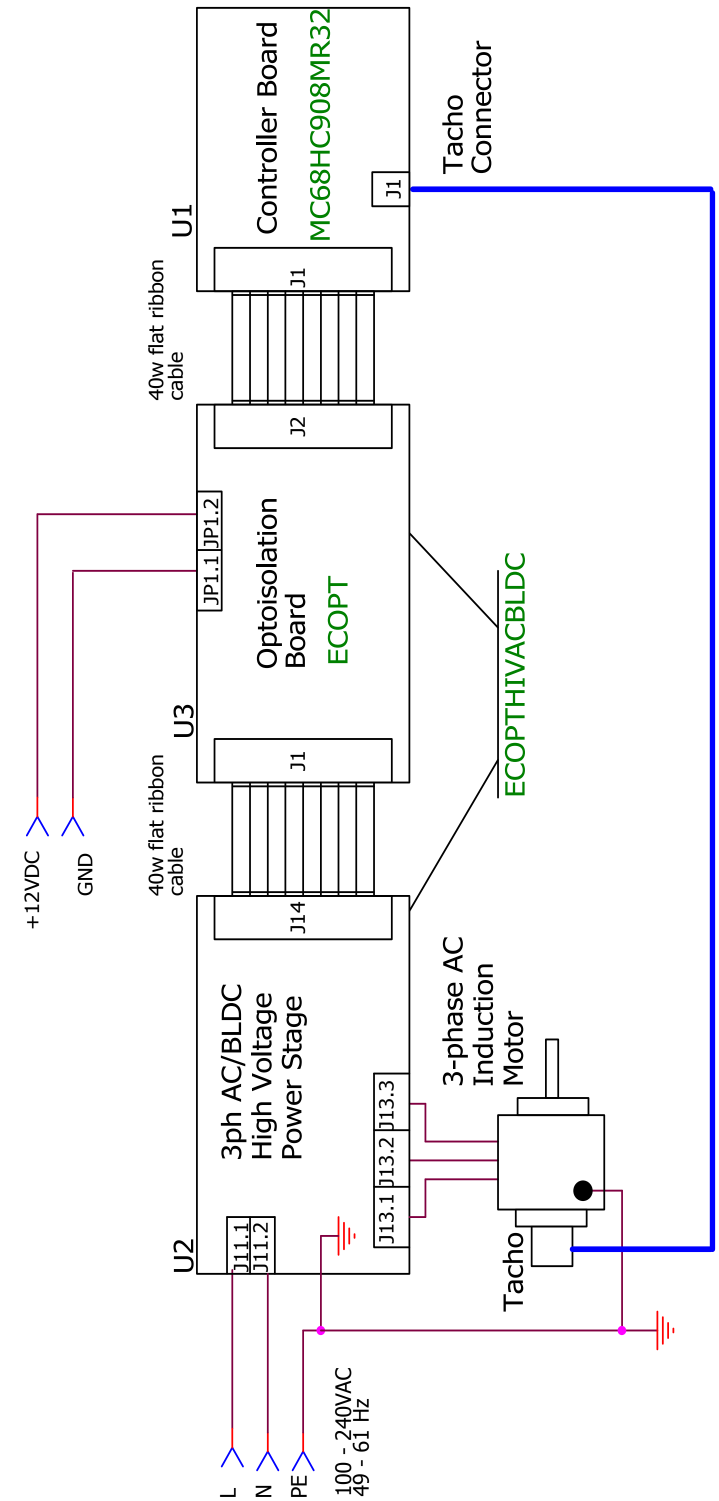

Figure 3-1. Hardware Configuration

3.3 MC68HC908MR32 Control Board

Motorola's embedded motion control series MR32 motor control board is designed to provide control signals for 3-phase AC induction, 3-phase brushless DC (BLDC), and 3-phase switched reluctance (SR) motors. In combination with one of the embedded motion control series power stages, and an optoisolation board, it provides a software development platform that allows algorithms to be written and tested without the need to design and build hardware. With software supplied on the CD-ROM, the control board supports a wide variety of algorithms for AC induction, SR, and BLDC motors. User control inputs are accepted from START/STOP, FWD/REV switches, and a SPEED potentiometer located on the control board. Alternately, motor commands can be entered via a PC and transmitted over a serial cable to DB-9 connector. Output connections and power stage feedback signals are grouped together on 40-pin ribbon cable connector. Motor feedback signals can be connected to Hall sensor/encoder connector. Power is supplied through the 40-pin ribbon cable from the optoisolation board or low-voltage power stage.

The control board is designed to run in two configurations. It can be connected to an M68EM08MR32 emulator via an M68CBL08A impedance matched ribbon cable, or it can operate using the daughter board. The M68EM08MR32 emulator board may be used in either an MMDS05/08 or MMEVS05/08 emulation system.

Figure 3-2 shows a block diagram of the board's circuitry.

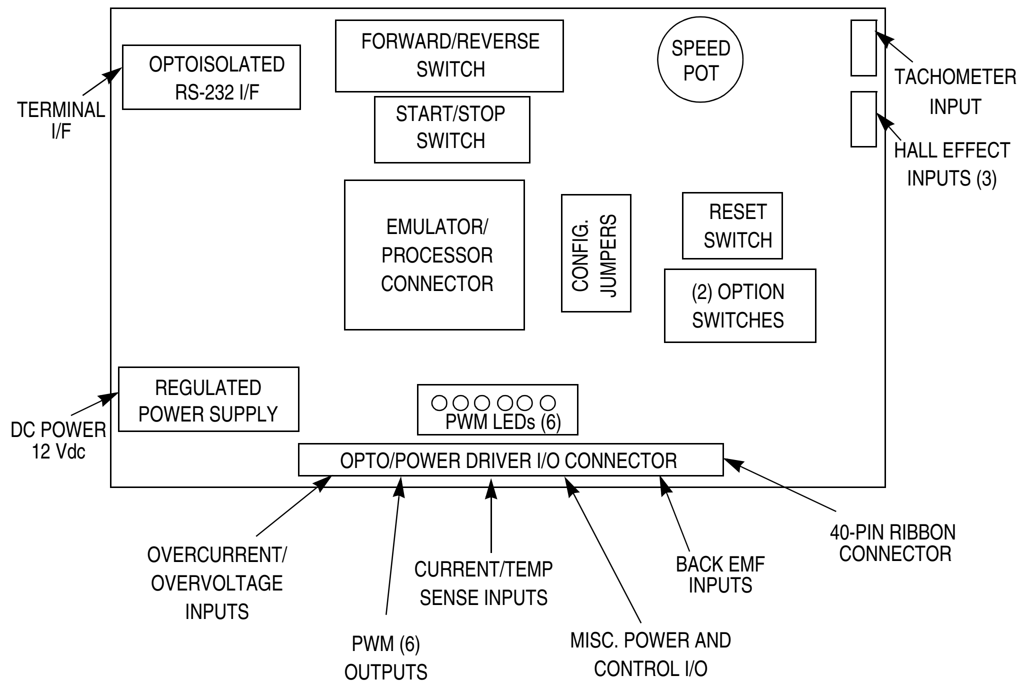

Figure 3-2. MC68HC908MR32 Control Board

The electrical characteristics in **Table 3-1** apply to operation at 25°C.

Table 3-1. Electrical Characteristics of Control Board

Characteristic	Symbol	Min	Typ	Max	Units
DC power supply voltage(1)	Vdc	10.8	12	16.5	V
Quiescent current	I_{CC}	—	80	—	mA
Min logic 1 input voltage (MR32)	V_{IH}	2.0	—	—	V
Max logic 0 input voltage (MR32)	V_{IL}	—	—	0.8	V
Propagation delay (Hall sensor/encoder input)	t_{dly}	—	—	500	ns
Analog input range	V_{In}	0	—	5.0	V
RS-232 connection speed		—	—	9600	Baud
PWM sink current	I_{PK}	—	—	20	mA

1. When operated and powered separately from other Embedded Motion Control tool set products

3.4 3-Phase AC BLDC High Voltage Power Stage

Motorola's embedded motion control series high-voltage (HV) AC power stage is a 180-watt (one-fourth horsepower), 3-phase power stage that will operate off of DC input voltages from 140 to 230 volts and AC line voltages from 100 to 240 volts. In combination with one of the embedded motion control series control boards and an optoisolation board, it provides a software development platform that allows algorithms to be written and tested without the need to design and build a power stage. It supports a wide variety of algorithms for both AC induction and brushless DC (BLDC) motors.

Input connections are made via 40-pin ribbon cable connector J14. Power connections to the motor are made on output connector J13. Phase A, phase B, and phase C are labeled PH_A, Ph_B, and Ph_C on the board. Power requirements are met with a single external 140- to 230-volt DC power supply or an AC line voltage. Either input is supplied through connector J11. Current measuring circuitry is set up for 2.93 amps full scale. Both bus and phase leg currents are measured. A cycle-by-cycle over-current trip point is set at 2.69 amps.

The high-voltage AC power stage has both a printed circuit board and a power substrate. The printed circuit board contains IGBT gate drive circuits, analog signal conditioning, low-voltage power supplies, power factor control circuitry, and some of the large, passive, power components. All of the power electronics which need to dissipate heat are mounted on the power substrate. This substrate includes the power IGBTs, brake resistors, current sensing resistors, a power factor correction MOSFET, and temperature sensing diodes. **Figure 3-3** shows a block diagram.

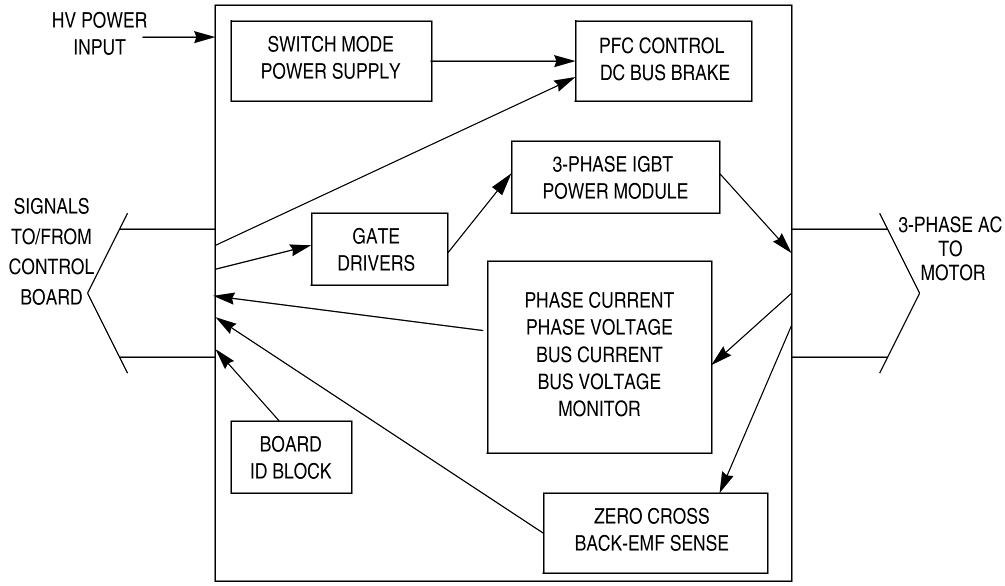

Figure 3-3. 3-Phase AC High Voltage Power Stage

The electrical characteristics in **Table 3-2** apply to operation at 25°C with a 160-Vdc power supply voltage.

Table 3-2. Electrical Characteristics of Power Stage

Characteristic	Symbol	Min	Typ	Max	Units
DC input voltage	Vdc	140	160	230	V
AC input voltage	Vac	100	208	240	V
Quiescent current	I_{CC}	—	70	—	mA
Min logic 1 input voltage	V_{IH}	2.0	—	—	V
Max logic 0 input voltage	V_{IL}	—	—	0.8	V
Input resistance	R_{In}	—	10 kΩ	—	
Analog output range	V_{Out}	0	—	3.3	V
Bus current sense voltage	I_{Sense}	—	563	—	mV/A
Bus voltage sense voltage	V_{Bus}	—	8.09	—	mV/V
Peak output current	I_{PK}	—	—	2.8	A
Brake resistor dissipation (continuous)	P_{BK}	—	—	50	W
Brake resistor dissipation (15 sec pk)	$P_{BK(Pk)}$	—	—	100	W
Total power dissipation	P_{diss}	—	—	85	W

3.5 Optoisolation Board

Motorola's embedded motion control series optoisolation board links signals from a controller to a high-voltage power stage. The board isolates the controller, and peripherals that may be attached to the controller, from dangerous voltages that are present on the power stage. The optoisolation board's galvanic isolation barrier also isolates control signals from high noise in the power stage and provides a noise-robust systems architecture.

Signal translation is virtually one-for-one. Gate drive signals are passed from controller to power stage via high-speed, high dV/dt, digital optocouplers. Analog feedback signals are passed back through HCNR201 high-linearity analog optocouplers. Delay times are typically

250 ns for digital signals, and 2 µs for analog signals. Grounds are separated by the optocouplers' galvanic isolation barrier.

Both input and output connections are made via 40-pin ribbon cable connectors. The pin assignments for both connectors are the same. For example, signal PWM_AT appears on pin 1 of the input connector and also on pin 1 of the output connector. In addition to the usual motor control signals, an MC68HC705JJ7CDW serves as a serial link, which allows controller software to identify the power board.

Power requirements for controller side circuitry are met with a single external 12-Vdc power supply. Power for power stage side circuitry is supplied from the power stage through the 40-pin output connector.

The electrical characteristics in **Table 3-3** apply to operation at 25°C, and a 12-Vdc power supply voltage.

Table 3-3. Electrical Characteristics of Optoisolation Board

Characteristic	Symbol	Min	Typ	Max	Units	Notes
Power Supply Voltage	Vdc	10	12	30	V	
Quiescent Current	I_{CC}	70[1]	200[2]	500[3]	mA	DC/DC converter
Min Logic 1 Input Voltage	V_{IH}	2.0	—	—	V	HCT logic
Max Logic 0 Input Voltage	V_{IL}	—	—	0.8	V	HCT logic
Analog Input Range	V_{In}	0	—	3.3	V	
Input Resistance	R_{In}	—	10	—	kΩ	
Analog Output Range	V_{Out}	0	—	3.3	V	
Digital Delay Time	t_{DDLY}	—	0.25	—	µs	
Analog Delay Time	t_{ADLY}	—	2	—	µs	

1. Power supply powers optoisolation board only.
2. Current consumption of optoisolation board plus DSP EVM board (powered from this power supply)
3. Maximum current handled by DC/DC converters

3.6 AC Induction Motor with Speed Tachogenerator

In the application a general purpose 4-pole AC induction motor is used. The 16-pole speed tachogenerator is coupled to the motor shaft. Output of the tachogenerator is the AC sinewave signal corresponding to the motor speed. It allows speed sensing of the motor required by the control algorithm. Detailed specifications are listed in **Table 3-4** In a target application a customer specific motor will be used.

Table 3-4. Motor Specifications

Motor Specification:	Motor Type:	Sg 71-4B 3-Phase AC Induction Motor
	Pole-Number:	4
	Nominal Speed:	1380 rpm
	Nominal Voltage:	3 x 220/380 V
	Nominal Power	370 W
	Nominal Current:	1.1 A
Speed Sensor	Type:	Speed Tachogenerator
	Pole-Number:	16

3.7 Sensors

The control algorithm requires speed and DC-Bus voltage sensing and DC-Bus over-current detection. Therefore, these sensors are built on the power stage board. Detailed schematics of the sensor circuits can be found in the user's manuals belonging to each board.

3.7.1 Speed Sensor

A 16-pole AC tachogenerator senses the actual speed of the motor. The output of the tachogenerator is an AC sinewave signal, its frequency corresponds to the motor speed. For a motor speed of 3000 rpm (100Hz synchronous) the tachogenerator output frequency is 400Hz (4-pole motor : 16-pole tachogenerator). The sinusoidal signal of the tachogenerator is filtered and transformed to a logic level square wave

by a squaring circuit. The generated square signal is fed to the microcontroller Input Capture block of the Timer A (Channel3). The Input Capture function reads the time between two subsequent rising edges of the generated square wave. The measured time corresponds to actual speed of the motor.

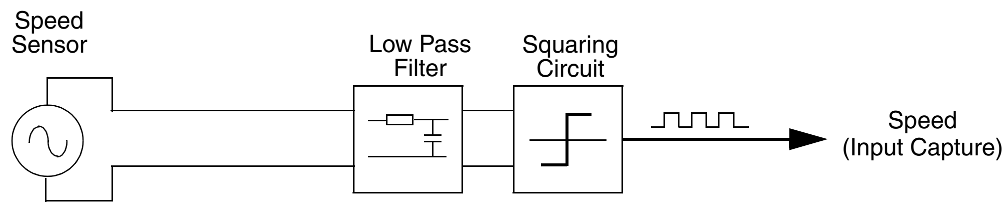

Figure 3-4. Speed Sensor Block Diagram

3.7.2 DC-Bus Voltage Sensor

The DC Bus voltage must be checked because of the over-voltage protection requirement.

A simple voltage sensor is created by a resistor divider. The voltage signal is transferred through the isolation amplifier and then amplified to the 5V reference level. The amplifier output is connected to the A/D converter of the microcontroller ATD1.

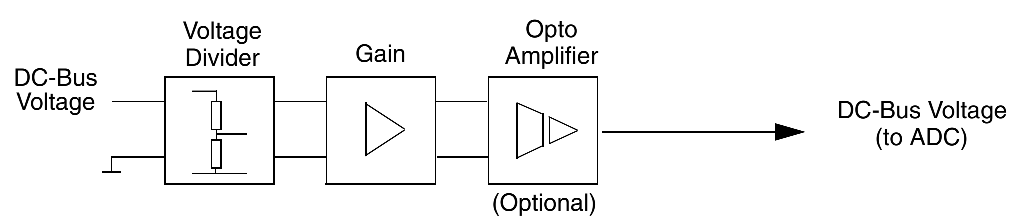

Figure 3-5. DC-Bus Voltage Sensor Block Diagram

3.7.3 DC-Bus Current Sensor

The DC-Bus current is checked because of the over-current protection requirement.

A current sensing resistor is inserted into the ground path of the DC-Bus lines. The ground of the drive is created on the inverter side of the sense resistor. This configuration advantage us that the voltage drop across the current sensing resistor has no influence on the gate driver signals. Because of this configuration, a positive DC-Bus current creates a negative voltage drop on the current sensing resistor. The voltage drop is amplified using an operational amplifier. The voltage signal is transferred through an optoisolation amplifier (optional).The measured DC-Bus current is compared with the threshold, and in case of over-current, a fault signal is generated. The fault signal is connected to the microcontroller fault input FAULT2.

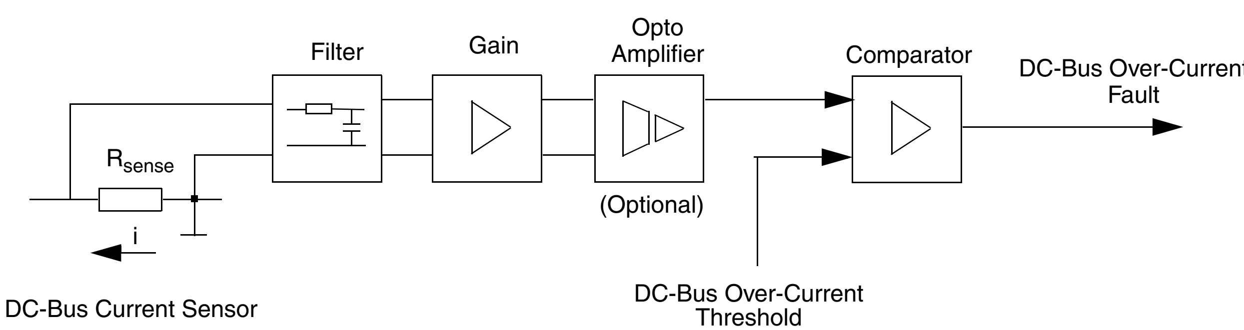

Figure 3-6. DC-Bus Current Sensor Block Diagram

3.8 Hardware Documentation

All the system parts are supplied and documented according to the following references:

- U1 - MC68HC908MR32 Control Board:
  - supplied as: ECCTR908MR32

  - described in: *Motorola Embedded Motion Control MC68HC908MR32 Control Board User's Manual* MEMCMR32CBUM/D

- U2 - 3 phase AC/BLDC High Voltage Power Stage
  - supplied in kit with Optoisolation Board as: ECOPTHIVACBLDC
  - described in: *Motorola Embedded Motion Control 3-Phase AC BLDC High-Voltage Power Stage User's Manual* MEMC3PBLDCPSUM/D
- U3 - Optoisolation Board
  - supplied with 3 ph AC/BLDC High Voltage Power Stage as: ECOPTHIVACBLDC
  - or supplied alone as: ECOPT - optoisolation board
  - described in: *Motorola Embedded Motion Optoisolation Board User's Manual* MEMCOBUM/D

Detailed descriptions of individual boards can be found in comprehensive User's Manuals belonging to each board. The manuals are available on the Motorola Web. The User's Manual incorporates the schematic of the board, description of individual function blocks and a bill of materials. An individual board can be ordered from Motorola as a standard product.

Section 4. Software Design

4.1 Contents

4.2 Introduction

This section describes the design of the software blocks of the drive. The software will be described in terms of:

- Control Algorithm Data Flow
- State Transition
- Software Listing
- Software Modifications for Open Loop Drive
- Microcontroller Memory and Peripheral Usage

4.3 Data Flow

The requirements of the drive dictate that the software takes some values from the user interface and sensors, processes them and generates 3-phase PWM signals for motor control.

The control algorithm of closed loop AC drive is described in **Figure 4-1** It consists of processes described in the following sub-sections. The special attention is given to the 3-phase PWM calculation subroutines. Also, initialisation of the microcontroller is described in a detail.

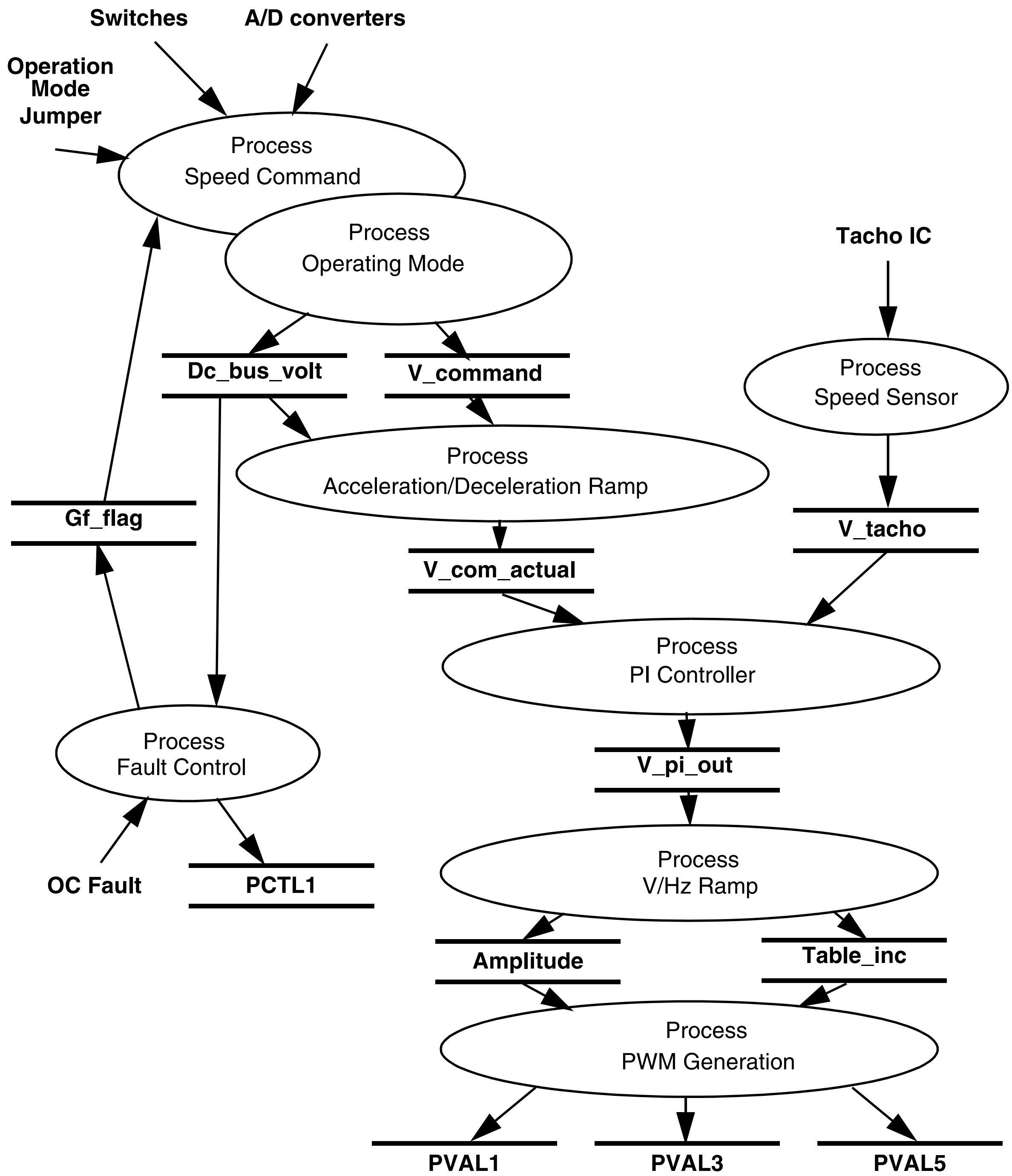

Figure 4-1. Data Flow

4.3.1 Speed Command and LED Control

The process has the following input parameters:

- DIP Operating Mode: Manual OM or Demo OM
  – DIP = OFF Demo Operating Mode
  – DIP = ON Manual Operating Mode
- Control Switches:
  – Start/Stop
  – Forward/Reverse
- A/D Converters:
  – potentiometer output for required speed
  – DC-Bus Voltage sensing
- General fault flag, `Gf_flag`

The process has the following output parameters:

- DC-Bus voltage, `Dc_bus_volt`
- Speed command, `V_command`

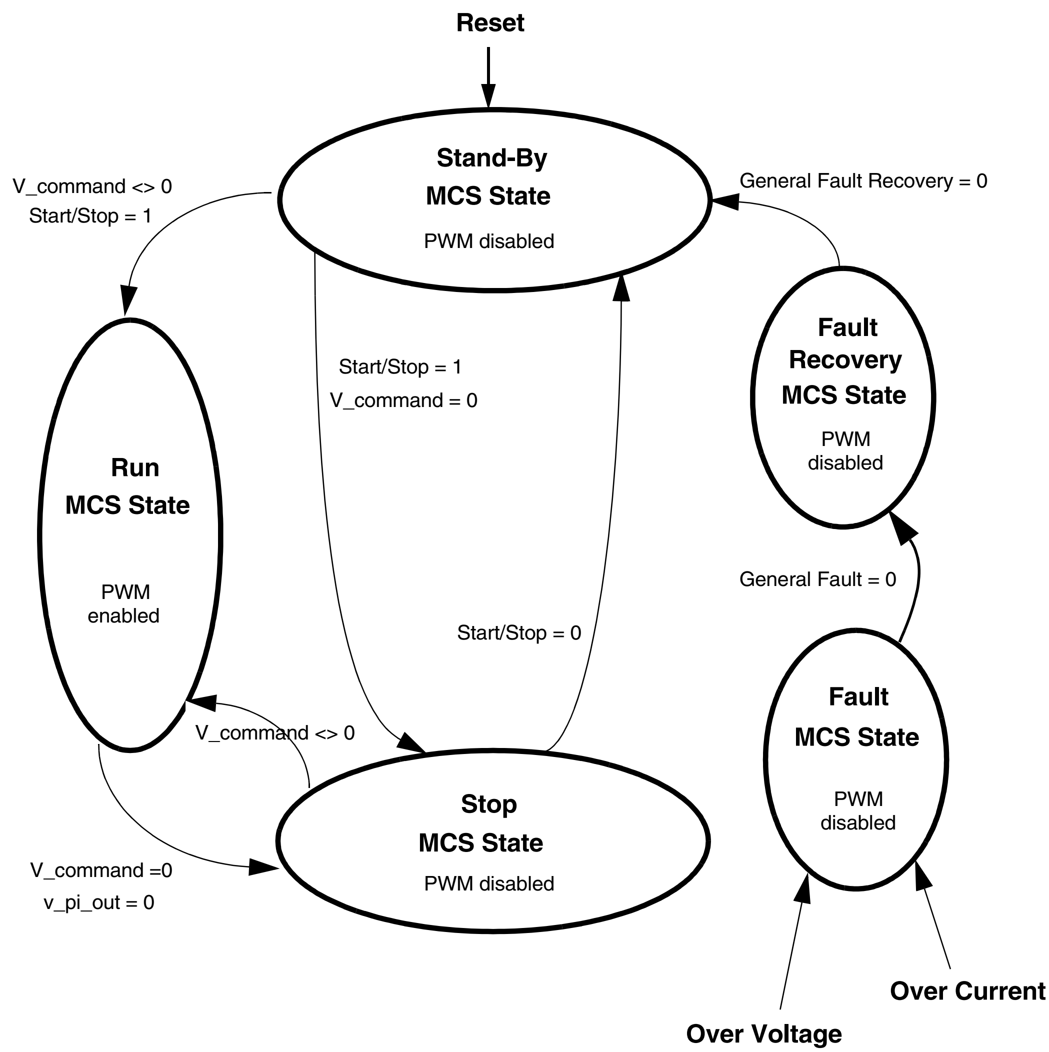

Figure 4-2. State Diagram of the Drive

The input parameters of the process are evaluated and the speed command, `V_command`, is calculated accordingly. Also the DC-Bus voltage, `Dc_bus_volt,` is measured. The general fault, `Gf_flag,` is analyzed and the state of the drive is set. The state diagram of the drive describes **Figure 4-2**. The status LED's are controlled according to the system state.

The calculated speed command, `V_command,` is a 2-byte variable with format 8.8 (1Hz = 0x10). This format is kept through all the program for all the speed variables.

4.3.2 Acceleration/Deceleration Ramp

The process calculates the new actual speed command based on the required speed according to the acceleration / deceleration ramp.

During deceleration the motor can work as a generator. In the generator state, the DC-Bus capacitor is charged and its voltage can easily exceed its maximal limit. Therefore the DC-Bus voltage is measured and compared with the limit. In case of deceleration over-voltage, the deceleration is interrupted and the motor runs with constant speed in order to discharge the capacitor. Then, the deceleration can continue.

4.3.3 Speed Sensor

The speed sensor process utilizes the IC function. It reads the time between the following rising edges of the speed sensor output and calculates the actual motor speed, `V_tacho`. Also, a software filter of the speed measurement can be incorporated in the process for better noise immunity. In this case the actual motor speed is calculated as average value of several measurements.

4.3.4 PI Controller

The speed closed loop control is characterized by a measurement of the actual motor speed. This information is compared with the reference set point and the error signal is generated. The magnitude and polarity of the error signal corresponds to the difference between actual and required speed. Based on the speed error, the PI controller generates the corrected motor frequency in order to compensate the error. The general principle of the speed PI control loop is illustrated in **Figure 4-3**.

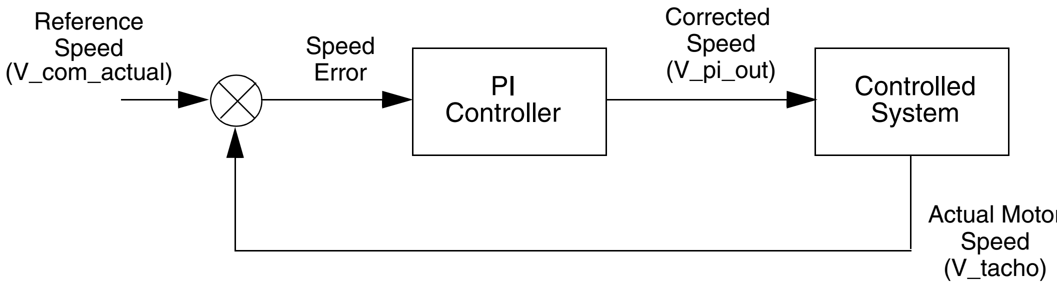

Figure 4-3. Closed Loop Control System

Process Description:

This process takes the input parameters: actual speed command, `V_com_actual`, and actual motor speed measured by a tachogenerator, `V_tacho`. It calculates a speed error and performs the speed PI control algorithm.

The output of the PI controller is a frequency of the first harmonic sine wave to be generated by the inverter: `V_pi_out`.

4.3.5 V/Hz Ramp

This process provides voltage calculation according to V/Hz ramp.

The input of this process is the generated inverter frequency, `V_pi_out`.

The outputs of this process are parameters required by PWM generation process:

- The table increment, `Table_inc`, that corresponds to the frequency, `V_pi_out`, and is used to roll through the wave table in order to generate the output inverter frequency
- Amplitude, `Amplitude`, of the generated inverter voltage

4.3.6 PWM Generation

This process generates a system of three phase sinewaves (or sinewaves with addition of third harmonic component) shifted 120^{o} each other.

The calculation is based on the wave table stored in ROM of the microcontroller. The table describes either a pure sinewave or sinewave with third harmonic addition. The second case is often preferred because it allows to generate a first harmonic sine voltage equal to the input AC line voltage. Because of sine symmetry only one quadrant of the wave period is stored in the table. The wave values for other quadrants are calculated from the first one. The format of the stored wave table data is from `#0x00` (for ZERO Voltage) up to PWM Modulus/2 (for the 100% Voltage). Thus the proper data scaling is secured.

It is important to note that 50% PWM (or 50% of PWM Modulus loaded to the corresponding PVAL registers) corresponds to the ZERO phase voltage. But in the wave table, the ZERO phase voltage corresponds to the number `#0x00`. Therefore, the fetched wave value from the table must be added to the 50% PWM Modulation for quadrant 1 and 2 or substracted from the 50% PWM Modulation for quadrant 3 and 4 (see point 5 of the process description). Thus the correct PWM value is loaded.

The input parameters of the process are:

- The table increment, `Table_inc,` that is used for the wave pointer update
- Amplitude, `Amplitude,` of the generated inverter voltage

The output parameters of the process are:

- PWM value for phase A: PVAL1 register
- PWM value for phase B: PVAL3 register
- PWM value for phase C: PVAL5 register

The process can be described by following points:

Phase A

1. Wave pointer for phase A is updated by the Table Increment
2. Based on the wave pointer of the phase the required wave quadrant is selected
3. The quadrant pointer is calculated from the wave pointer with respect to the related quadrant
4. Table value determined by quadrant pointer is fetched from the wave table
5. The table value is added to (or substracted from) the 50% modulus with respect to the related quadrant
6. The result is loaded to the PVAL1 register; PVAL2 register is loaded automatically because of complementary PWM mode selected during the PWM module initialisation

Phase B

1. The phase B wave pointer is calculated as phase A wave pointer + 1/3 of wave period (1/3 of `0xffff` equals to `0x5555`)
2. See corresponding points 2.-5. of the Phase A calculation
3. The result is loaded to the PVAL3 register; PVAL4 register is loaded automatically because of complementary PWM mode

Phase C

1. The phase C wave pointer is calculated as phase A wave pointer + 2/3 of wave period (2/3 of `0xffff` equals to `0xaaaa`)
2. See corresponding points 2.-5. of the Phase A calculation
3. The result is loaded to the PVAL5 register; PVAL6 register is loaded automatically because of complementary PWM mode

The process is accessed regularly in the rate given by the set PWM frequency and the selected PWM interrupt prescaller (register `PCTL2`). This process has to be repeated often enough compared to the wave frequency in order to generate the correct wave shape. Therefore for 16kHz PWM frequency, it is called each fourth PWM pulse and thus the PWM registers are updated in 4kHz rate (each 250µsec).

4.3.7 Fault Control

This process is responsible for fault handling. The software accommodates two fault inputs: DC-Bus over-current and DC-Bus over-voltage.

DC-Bus over-current: In case of over-current, the external hardware provides a rising edge on the fault input of the microcontroller FAULT2. This signal disables all motor control PWM's outputs (PWM1 - PWM6) and sets general fault flag, `Gf_flag`.

DC-Bus over-voltage: The sensed DC-Bus voltage is compared with the limit within the software. In case of over-voltage all motor control PWM outputs are disabled (PCTL1) and the general fault flag, `Gf_flag,` is set.

If any of the faults occurs, the recovery time for the individual fault is loaded and till this time expires, the system remains disabled.

4.4 Software Implementation

The processes described above are implemented in a single state machine, as illustrated in **Figure 4-4**, **Figure 4-5** and **Figure 4-6**.

The general software implementation incorporates the main routine entered from Reset and three interrupt states. The Main Routine includes the initialisation of the microcontroller and a Software Timer for the control algorithm time base. The interrupt states provide calculation of actual speed of the motor, over-current fault handler and PWM generation process.

4.4.1 Initialisation

The Main Routine provides initialisation of the microcontroller:

- clears RAM
- initialises PLL Clock
- initialises PWM module:
  - center aligned complementary PWM mode, positive polarity (MOR register)
  - COP and LVI enable (MOR register)
  - PWM modulus - defines the PWM frequency (PMOD register)
  - 2µsec dead time (DEADTM register)
  - PWM interrupt reload every fourth PWM pulse (PCTL2 register)
  - FAULT2 (over current fault) in manual mode, interrupt enabled (FCR register)
- sets up I/O ports
- initialises Timer B for IC and for software timer reference
- initialises Analog to Digital Converter
- sets up Operating Mode (Manual OM or Demo OM)
- enables interrupts

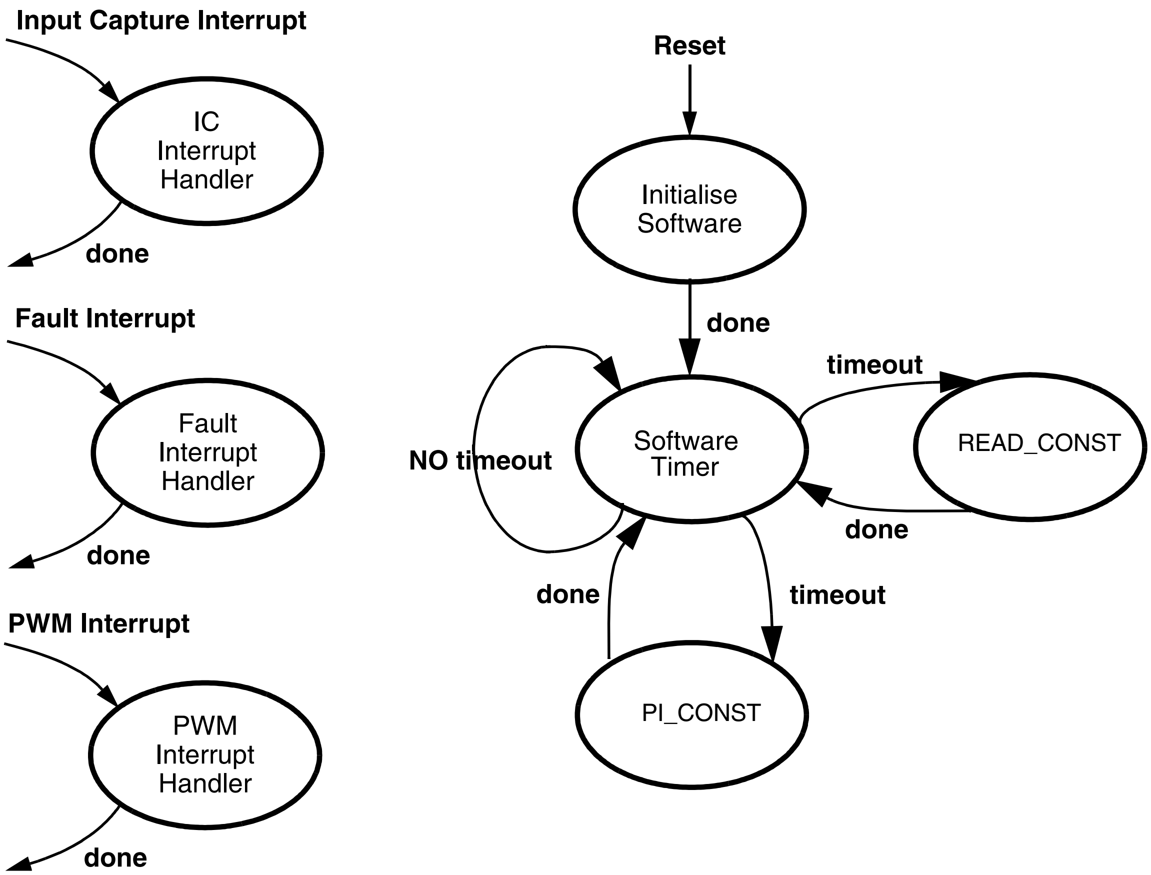

Figure 4-4. Software Implementation - General Overview

An example of initialisation of PLL Clock and Motor Control PWM Modules for MC68MC908MR32 is the following:

```
        /* setup PLL clock */

PBWC = 0x80;                     /* set Auto Bandwidth Control */
while (~PBWC & 0x40);            /* wait for PLL lock */
PCTL = 0x30;                     /* use PLL clock */

        /* setup Motor Control PWM module */

MOR = 0x00;    /* 0x00: pos. center PWM; COP and LVI enabled */
               /* 0x60: neg. center PWM; COP and LVI enabled */
```

```
PMOD = 0x00e6;          /* set up PWM modulus => PWM frequency */
                        /* for 7.3728MHz Bus Frequency
                        PWM_MODULUS = 0x00e6 gives 16kHz PWM */
DEADTM=15;              /* 2usec deadtime = #15
                        (for Bus freq. = 7.3728MHz) */
DISMAP=0xff;            /* when PWM disabled, disable PWM1-6 */
PCTL2 = 0x80;           /* PWM interrupt every 4th. PWM loads */
PCTL1 |= 0xc0;          /* disable MCPWM */
PWMOUT = 0x00;          /* output port control is PWM gen. */
PCTL1 |= 0x02;          /* set LDOK bit */
FCR |= 0x08;            /* Flt2 enabled in manual mode */

PVAL1 = PWM_MODULUS/2; /* set phase A pwm to 50% */
PVAL3 = PWM_MODULUS/2; /* set phase B pwm to 50% */
PVAL5 = PWM_MODULUS/2; /* set phase C pwm to 50% */
```

When all modules of the microcontroller are initialised, enable the PWM module:

```
PCTL1 |= 0x20;          /* enables PWM interrupts */
PCTL1 |= 0x01;          /* enables PWM */
```

4.4.2 Software Timer

The software timer routine provides the timing sequence for required subroutines. The software timer is performed instead of an Output Capture interrupt handler because of lack of interrupt priority in the HC08 MCU. The main program has several time-demanding interrupt routines and more interrupt requirements can cause a software fault.

The software timer routine has two timed outputs -

- in `READ_CONST` timeout, there is a routine that scans inputs, calculates speed command, handles fault routines and the LED driver

- in PI_CONST timeout, there is a routine that provides over-voltage protection during deceleration, speed ramp (acceleration/deceleration), PI controller, V/Hz ramp and provides parameters for PWM generation

The interrupt handlers have the following functions:

- Input Capture Interrupt Handler reads the time between the two subsequent IC edges (basic part of the Process Speed Sensor)
- Fault Interrupt Handler takes care of over-current fault interrupt (over-current part of the Process Fault Control)
- PWM Interrupt Handler generates system of three-phase voltages for the motor (Process PWM Generation)

4.4.3 READ_CONST Timeout State

This state is accessed from the main software timer in READ_CONST rate. The following sequence is performed (see **Figure 4-5**):

- All the inputs are scanned (DC-Bus voltage, speed pot, Start/Stop switch, Forward/Reverse switch)
- According to the operating mode, the speed command is calculated
- The DC-Bus voltage is compared with the over-voltage limit. Also, over-current fault flag is checked
- In case of a fault, the fault recovery routine is entered and till the recovery time expires, the drive stays disabled
- Finally, the LED driver controls individual LEDs according to the status of the drive

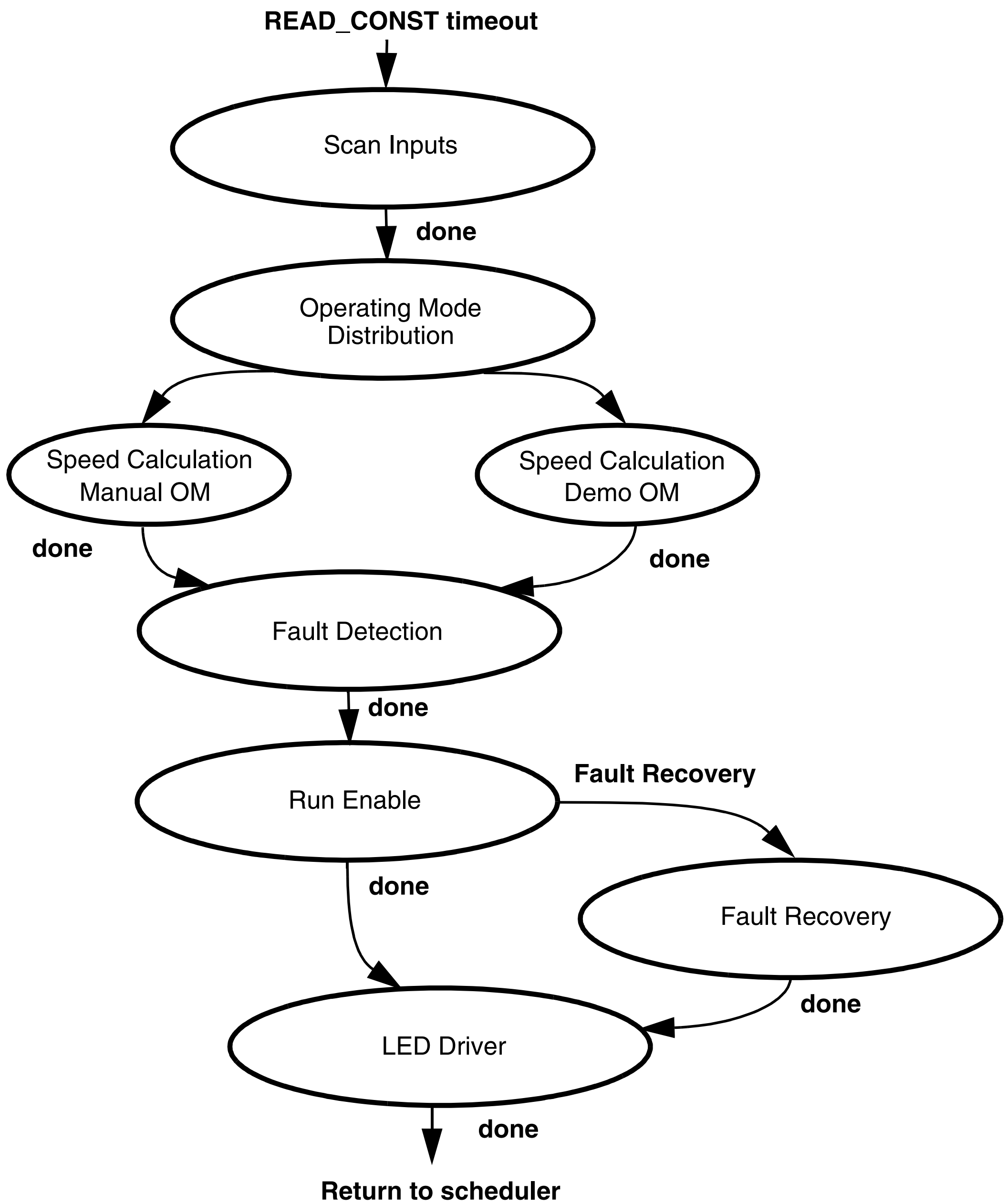

Figure 4-5. READ_CONST Timeout Routine

4.4.4 PI_CONST Timeout Routine

This routine is accessed from the main software timer in PI_CONST rate. The rate defines the time constant of the PI controller. The following sequence is performed (see **Figure 4-6**):

- During deceleration, the DC-Bus voltage is checked and in case of deceleration over-voltage, the deceleration is interrupted until the capacitor is discharged,
- When no deceleration over-voltage is measured, the acceleration/deceleration speed profile is calculated,
- Actual motor speed is calculated,
- PI speed controller is performed and the corrected motor frequency calculated,
- The corresponding voltage amplitude is calculated according to the Volt-per-Hertz ramp. Thus both parameters for PWM generation are available (`Table_inc, Amplitude`).

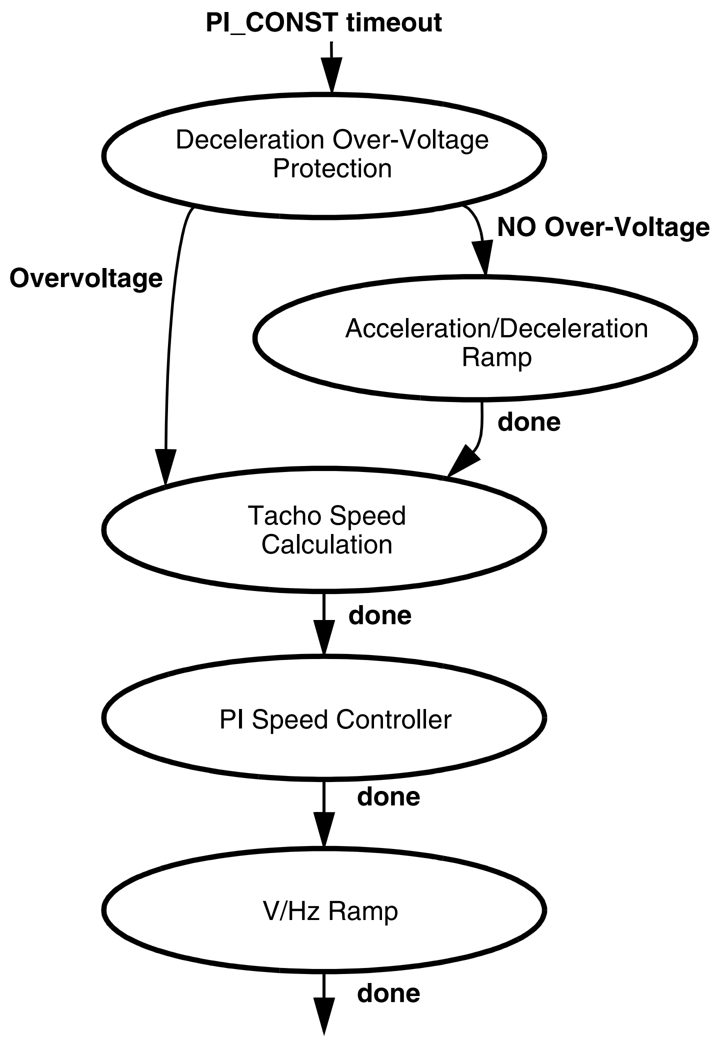

Figure 4-6. PI_CONST Timeout Routine

4.5 Software Listing

The software listing is also available for this application. Special attention was given to the modularity of the code. The code is written in C (Metrowerks CodeWarrior® for MC68HC08 microcontrollers).

The software consists of the following parts:

- MAIN.C

  It is the entry point following a Reset. It contains the Initialize Software Routine code, the Main state machine with the Software Timer.
- SPEED.C

  It contains READ_CONST Timeout code (Scan Inputs, OM Distribution , Speed Calculation - Manual OM, Speed Calculation - Demo OM, Fault Detection, Run Enable, Fault Recovery, LED Driver).
- RAMP.C

  It contains code for ramps: Acceleration/Deceleration Ramp, V/Hz Ramp.
- PI.C

  It contains PI_CONST Timeout code (Deceleration Over-voltage Protection, Tacho Speed Calculation, PI Speed Controller and calls Acceleration/Deceleration Ramp and V/Hz Ramp appropriately).
- FAULT.C

  It contains Fault Interrupt Handler code.
- PWMCALC.C

  It contains PWM Calculation Interrupt Handler code.
- TACHO.C

  It contains Tacho Interrupt Handler code.
- 3RDHQUAD.H

The header file contains the first quadrant of sinewave with its 3rd. harmonic injection - 256 unsigned 2-byte entries.

- RAM.H

  It contains the global RAM variable definitions for the whole project.

- CONST.H

  It contains the global constant definitions for the whole project.

- VECTORS.H

  It contains the interrupt vectors.

4.6 Open Loop Drive

The system presented in this application note can also run in an open loop mode. In this case, the actual motor speed is not measured and the generated voltage frequency directly corresponds to the externally set speed command and is not corrected by any controller according to the actual motor speed.

Because the motor is asynchronous, the actual motor speed varies with the mechanical motor load. The higher mechanical load the higher slip of the motor and the lower motor speed. Therefore, the speed precision of the drive is not so high. For some application, such behaviour of the drive is not acceptable (like washing machine), some other can withstand it. An example of the application can be a fan, a compressor, a pump, etc., where performance of the open loop drive is sufficient. The advantage of the open loop drive is its relative simplicity of both hardware and software design compared to the closed loop system.

The open loop system design has the following modifications:

The hardware design doesn't require the speed transducer and speed sensing circuitry.

The software for Open Loop drive requires the following modifications (see **Figure 4-2**):

- Remove Process PI Controller
- Remove Process Speed Sensor and disable IC Interrupt
- Load an output of the Process Acceleration/Deceleration ramp to an input of the Process Volt-per-Hertz ramp (Set variable `V_out = V_com_actual`)

In the provided software, the open loop control can be set during the software initialisation:

```
Speed_control = OPEN_LOOP; /* for open speed control loop */
```

or

```
Speed_control = CLOSED_LOOP; /* for closed speed control loop */
```


Freescale Semiconductor, Inc.

4.7 Microcontroller Usage

Table 4-1 shows how much of memory is needed to run the 3-phase AC drive in the speed closed loop. A significant part of the microcontroller memory is still available for other tasks.

Table 4-1. Memory Usage

Memory	Available (MC68HC908MR32)	Used
FLASH	**32 kBytes**	**3.7 kBytes**
RAM	**768 Bytes**	**82 Bytes**

The MC68HC908MR32 microcontroller offers many features that simplify the drive design. The following table describes individual available blocks and their usage for the introduced system.

Table 4-2. MR32 Modules Usage

Module available on MC68HC908MR32	Used	Purpose
PWMMC	yes	3-phase PWM generation, fault protection
Timer A (4-channel)	yes	Time base for control algorithm (TACNT), Input Capture for measurement of actual motor speed (TA3)
Timer B (2-channel)	no	-
SPI	no	-
SCI	no	-
I/O ports	yes	User interface, LEDs
COP	yes	S/W runaway protection
IRQ	no	-
LVI	yes	Low voltage protection
ADC	yes	Speed set-up DC-Bus voltage measurement
POR	yes	Reset after Power ON

Section 5. System Setup

5.1 Contents

5.2 Hardware Setup

Figure 5-1 illustrates the hardware setup for the application. It incorporates the following modules:

- MC68HC908MR32 Control Board
- 3-phase AC/BLDC High Voltage Power Stage
- Optoisolation Board
- 3-phase AC induction motor with speed tachogenerator

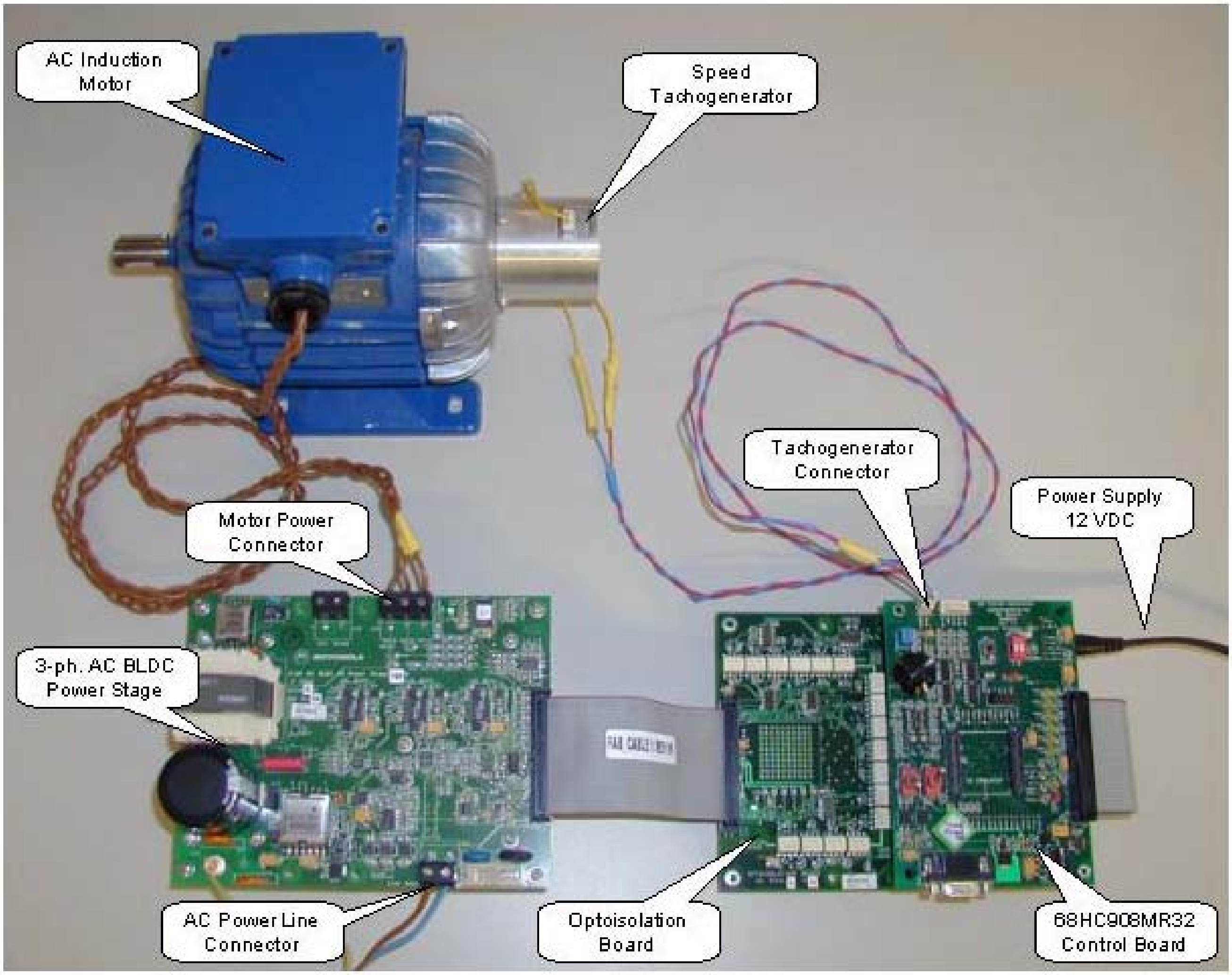

Figure 5-1. Setup of the Application

5.3 Warning

This application operates in an environment that includes dangerous voltages and rotating machinery.

Be aware, that the application power stage and optoisolation board are not electrically isolated from the mains voltage - they are live with risk of electric shock when touched.

An isolation transformer should be used when operating off an AC power line. If an isolation transformer is not used, power stage grounds and

oscilloscope grounds are at different potentials, unless the oscilloscope is floating. Note, that probe grounds and, therefore, the case of a floated oscilloscope are subjected to dangerous voltages.

The user should be aware, that:

- Before moving scope probes, making connections, etc., it is generally advisable to power down the high-voltage supply.
- To avoid inadvertent touching live parts, use plastic covers.
- When high voltage is applied, using only one hand for operating the test setup minimizes the possibility of electrical shock.
- Operation in lab setups that have grounded tables and/or chairs should be avoided.
- Wearing safety glasses, avoiding ties and jewelry, using shields, and operation by a personnel trained in high-voltage lab techniques is also advisable.
- Power transistors, the PFC coil, and the motor can reach temperatures hot enough to cause burns.
- When powering down; due to storage in the bus capacitors, dangerous voltages are present until the power-on LED is off.

5.4 Jumper Settings of Controller Board

The MC68HC908MR32 control board jumper settings shown in **Figure 5-2** and **Table 5-1** are required to execute the 3-phase AC motor control application with tachogenerator. For a detailed description of the jumper settings, refer to the *MC68HC908MR32 Control Board User's Manual* (Motorola document order number MEMCMR32CBUM/D).

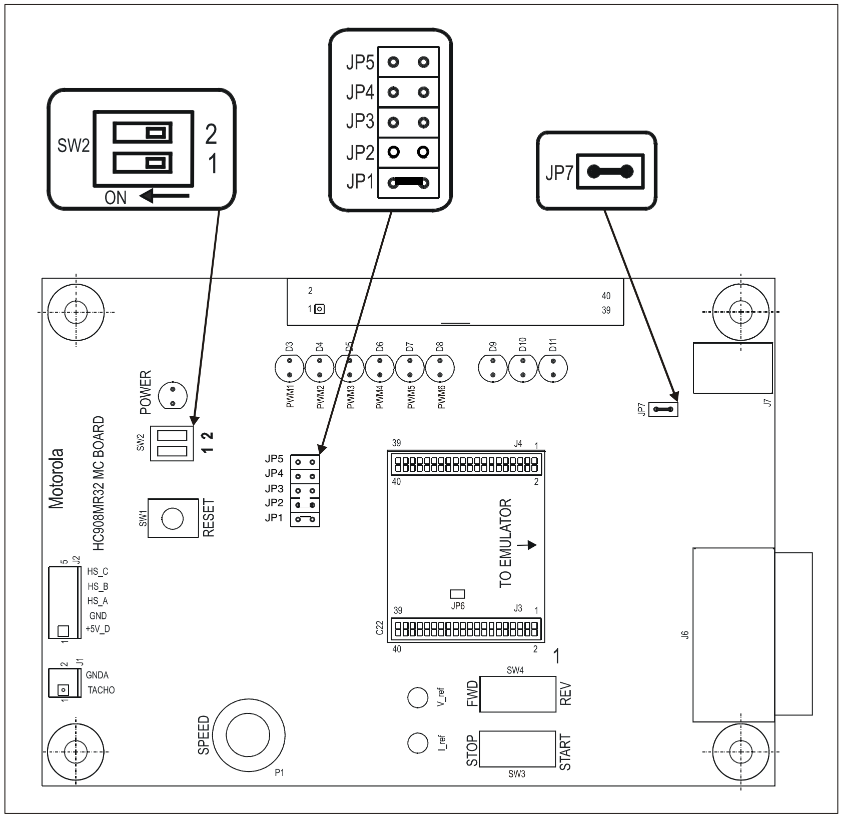

Figure 5-2. MC68HC908MR32 Jumper Reference

Table 5-1. MC68HC908MR32EVM Jumper Settings

Jumper Group	Comment	Connections
JP1	Tachometer input selected	1–2
JP2	Encoder input selected	No connection
JP3	Back EMF signals selected	No connection
JP4	Power factor correction — zero cross signal selected	No connection
JP5	Power factor correction — PWM signal selected	No connection
JP7	Power Supply connected to jack J3	1–2

5.5 Required Software Tools

The application requires the following software development tools:

- Metrowerks[1] CodeWarrior®[2] for MC68HC08 microcontrollers version 1.2 or later.
- PC master software version 1.2.0.11 or later

5.6 Building the Application

To build this application, open the ***3ph_acim_vhz.mcp*** project file and execute the *Make* command; see **Figure 5-3** This command will build and link the motor control application along with all needed Metrowerks libraries.

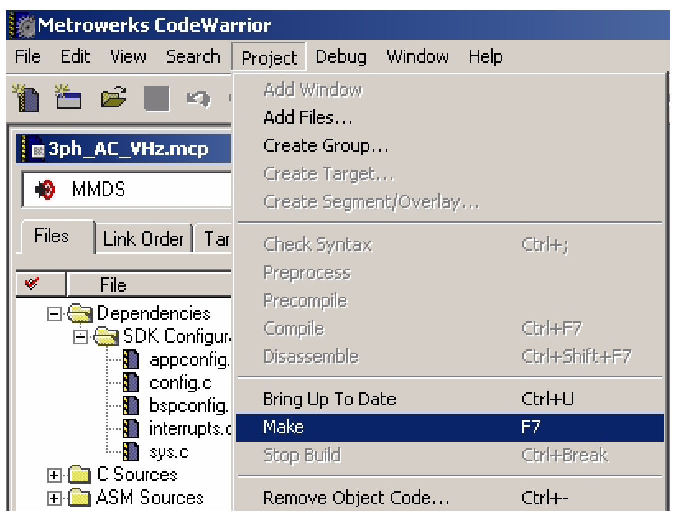

Figure 5-3. Execute *Make* Command

1. Metrowerks® and the Metrowerks logo are registered trademarks of Metrowerks, Inc., a wholly owned subsidiary of Motorola, Inc.

2. CodeWarrior® is a registered trademark of Metrowerks, Inc., a wholly owned subsidiary of Motorola, Inc.

5.7 Executing the Application

To execute the motor control application, choose the *Project/Debug* command in the CodeWarrior® IDE, followed by the *Run* command.

If the MMDS target is selected, CodeWarrior will automatically download to the MMDS05/08 emulator.

The application can operate in two modes:

1. Manual Operating Mode

   The drive is controlled by the START/STOP switch (SW3). The direction of the motor rotation is set by the FWD/REV switch (SW4). The motor speed is set by the SPEED potentiometer (P1). Refer to **Figure 5-4** for this description.

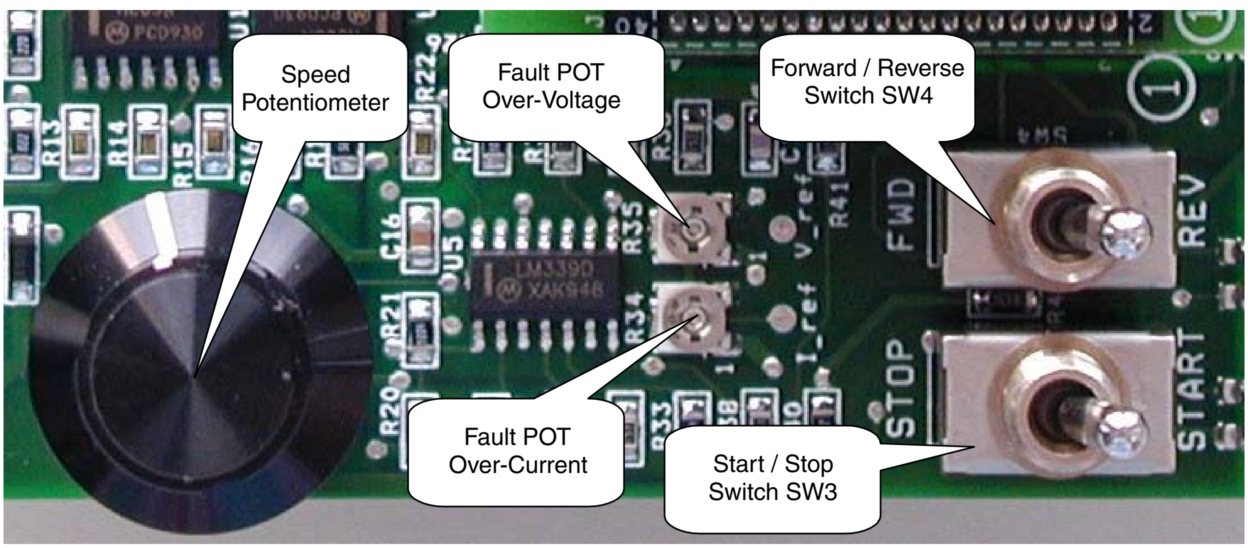

Figure 5-4. Control Elements

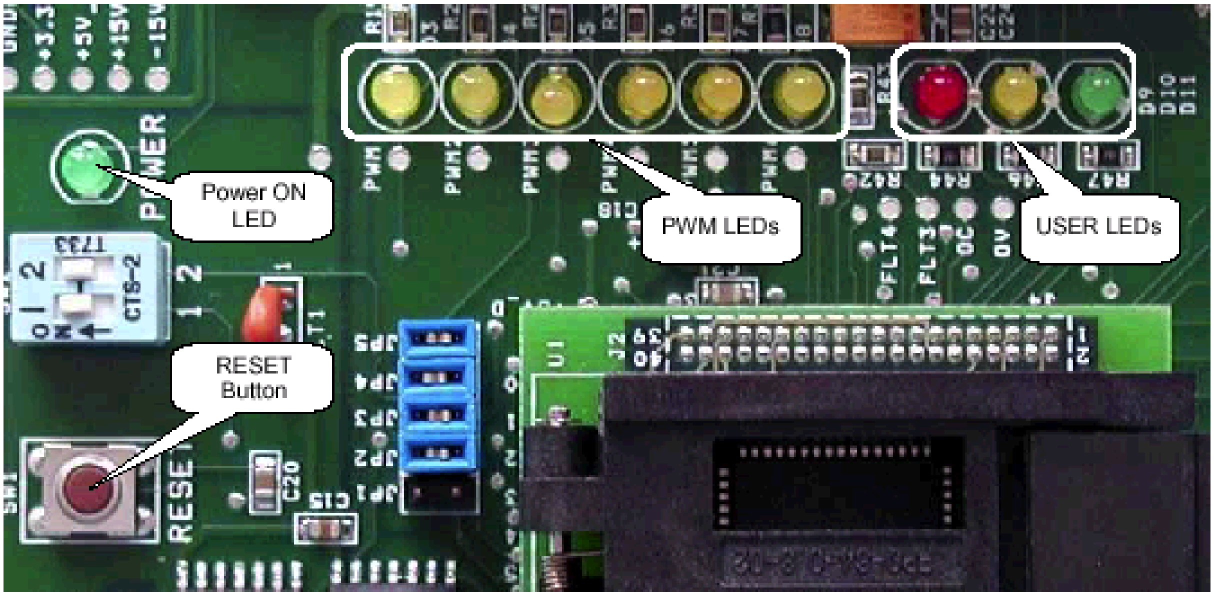

Figure 5-5. USER LEDs, PWM LEDs, and RESET

2. Demo Operating Mode

   In the Demo Operating Mode, the required speed profile is pre-programmed and the only control input is the switch “Start“. The pre-programmed profile can be changed in the s/w. The drive is enabled by the START/STOP switch, which can be used to safely stop the application at any time.

The application states are displayed with on-board LEDs. If the application runs and motor spinning is disabled (i.e., the system is ready), the yellow status LED will be on. When motor rotation is enabled, the green status LED will be on, and the actual state of the pulse-width modulator (PWM) outputs are indicated with PWM output LEDs, labeled PWM1 - PWM6. If DC-Bus over-current / DC-Bus over-voltage occurs the red fault LED will be turned on. This fault state can be exited when the fault condition disappears and the safety fault time-out expires.

Refer to **Table 5-2** for a description of the application states and to **Figure 5-5** for the on-board LEDs position.

Table 5-2. Motor Application States

Application State	Motor State	LED's State
Stand-By	Stopped	Yellow LED ON
Run	Spinning	Green LED ON
Fault	Stopped	Red LED ON

Once the application is running:

- Move the START/STOP switch (SW3) from STOP to START
- Select the direction of rotation by the FWD/REV switch (SW4)
- Set the required speed by the SPEED potentiometer

If successful, the 3-phase AC induction motor will be spinning.

NOTE: *If the START/STOP switch is set to the START position when the application starts, toggle the switch between the STOP and START positions to enable motor spinning. This is a protection feature preventing the motor to start spinning when the application is executed from CodeWarrior.*

Appendix A. References

1. Motorola, Inc. (2001). *68HC908MR32 User's Manual*, MC68HC908MR32/D
2. Motorola, Inc. (2000). *Motorola Embedded Motion Control MC68HC908MR32 Control Board User's Manual*, MEMCMR32CBUM/D
3. Motorola, Inc. (2000). *Motorola Embedded Motion Control 3-Phase AC BLDC High-Voltage Power Stage User's Manual*, MEMC3PBLDCPSUM/D
4. Motorola, Inc. (2000). *Motorola Embedded Motion Optoisolation Board User's Manual*, MEMCOBUM/D
5. Motorola, Inc. (1997). *Making Low-Distortion Motor Waveforms with the MC68HC708MP16,* AN1728

Appendix B. Glossary

AC — Alternating Current.

ACIM — AC Induction Motor.

ADC — analogue-to-digital converter.

A/D Converter — analogue-to-digital converter.

BLDC — brushless DC motor.

DC — Direct Current.

DT — see “Dead Time (DT)”

Dead Time (DT) — short time that must be inserted between the turning off of one transistor in the inverter half bridge and turning on of the complementary transistor due to the limited switching speed of the transistors.

duty cycle — A ratio of the amount of time the signal is on versus the time it is off. Duty cycle is usually represented by a percentage.

interrupt — A temporary break in the sequential execution of a program to respond to signals from peripheral devices by executing a subroutine.

input/output (I/O) — Input/output interfaces between a computer system and the external world. A CPU reads an input to sense the level of an external signal and writes to an output to change the level on an external signal.

logic 1 — A voltage level approximately equal to the input power voltage (V_{DD}).

logic 0 — A voltage level approximately equal to the ground voltage (V_{SS}).

MC68HC08 — A Motorola family of 8-bit MCUs.

MCS — Motor Control System

MCU - Microcontroller Unit. A complete computer system, including a CPU, memory, a clock oscillator, and input/output (I/O) on a single integrated circuit.

MR32 (908MR32) — Motorola MC68HC908MR32 microcontroller dedicated for motor control applications

phase-locked loop (PLL) — A clock generator circuit in which a voltage controlled oscillator produces an oscillation which is synchronized to a reference signal.

PVAL — PWM value register of motor control PWM module of MC68HC908MR32 microcontroller. It defines the duty cycle of generated PWM signal.

PWM — Pulse Width Modulation

reset — To force a device to the known condition.

SCI — See "serial communications interface module (SCI)"

serial communications interface module (SCI) — A module that supports asynchronous communication.

serial peripheral interface module (SPI) — A module that supports synchronous communication.

software (s/w) — Instructions and data that control the operation of a microcontroller.

software interrupt (SWI) — An instruction that causes an interrupt and its associated vector fetch.

SPI — See "serial peripheral interface module (SPI)."

SR — switched reluctance motor.

timer — A module used to relate events in a system to a point in time.

HOW TO REACH US:

USA/EUROPE/LOCATIONS NOT LISTED:

Motorola Literature Distribution;
P.O. Box 5405, Denver, Colorado 80217
1-303-675-2140 or 1-800-441-2447

JAPAN:

Motorola Japan Ltd.; SPS, Technical Information Center,
3-20-1, Minami-Azabu Minato-ku, Tokyo 106-8573 Japan
81-3-3440-3569

ASIA/PACIFIC:

Motorola Semiconductors H.K. Ltd.;
Silicon Harbour Centre, 2 Dai King Street,
Tai Po Industrial Estate, Tai Po, N.T., Hong Kong
852-26668334

TECHNICAL INFORMATION CENTER:

1-800-521-6274

HOME PAGE:

http://motorola.com/semiconductors

MOTOROLA

DRM020/D